ECE 265
Homework #6

Problems:

1. Hand assemble the following program by providing the address and contents for the data and instructions.

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>CONTENTS (HEX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.</td>
<td></td>
</tr>
<tr>
<td>* A Program to Display the Analog</td>
<td></td>
</tr>
<tr>
<td>* Input on a Bar Graph</td>
<td></td>
</tr>
<tr>
<td>*</td>
<td></td>
</tr>
</tbody>
</table>

STACK EQU $00FF
PORTB EQU $1004
ADCTL EQU $1030
ADR1 EQU $1031
OPTION EQU $1039

ORG $E000
MAIN: LDS #STACK
LDA #$80
STAA OPTION
LDA #$20
STAA ADCTL
LOOP: JSR BAR_GRAPH
BRA LOOP

ORG $F000
BAR_GRAPH: PSHX
PSHA
PSHB
TPA
PSHA
LDX #TABLE
LDAB #$FF
LDA ADR1
2. For the program of Problem 1, what are the contents of the following registers and stack locations after execution of the LDAB #$FF instruction, the first time? Assume that the initial value of CCR is $D0.

<table>
<thead>
<tr>
<th>Register or stack location</th>
<th>Contents (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
</tr>
<tr>
<td>top byte of stack</td>
<td></td>
</tr>
<tr>
<td>sixth byte in stack</td>
<td></td>
</tr>
</tbody>
</table>

3. If the ASLD instruction were not available in the M68HC11, what two instructions together would accomplish this?

4. Write a subroutine to multiply the signed, 2's complement number in the D register by 10 leaving the result in D. Assume no overflow will occur.

Note that the multiply instruction (MUL) cannot be used directly because it only works on 8-bit unsigned numbers. However, note that to multiply by 10 is to multiply by $(8+2)$. Also, note that a left shift of D, the appropriate number of times, can be used to multiply by any power of 2. Finally, use the stack for any temporary storage of data.
5. A list of numbers, starting at location LIST, is to be searched for a match of the number found in the A accumulator. Write a subroutine called PROCESS which on entry contains the number to be found in the A accumulator and the total number of elements in the list in the B register. On exit from the subroutine, the B register should contain the index of the match farthest from the top of the list, that is, nearest the end of the list. If no match is found, then B should contain -1 on return.

For this problem, save any registers that are needed in the operation, on the stack. Restore them before returning from the subroutine. That is, only the value of register B should change from calling the subroutine.

Hint: consider processing the LIST from the end back to the beginning (top).

6. Write a subroutine, called NIB_SWAP, that swaps the high-order and low-order nibbles of the contents of the A accumulator. For example, if the contents of A on entry to the subroutine is $5A$, then A will contain $A5$ on exit. Save and restore any registers used in the subroutine. Use the stack for any temporary storage of data.